Progetto N°1 - Calcolo Scientifico

Nodari Alessandro & Proserpio Lorenzo

April 2021

Problema 1

Soluzione analitica

Notiamo immediatamente che l'equazione può essere standardizzata nel modo seguente:

$$u''(x) = \sqrt{1 + (u'(x))^2} \Longrightarrow \begin{cases} u'(x) = v(x) \\ v'(x) = \sqrt{1 + (v(x))^2} \end{cases}$$

a questo punto si nota che la seconda equazione è a variabili separabili, quindi può essere integrata agilmente:

$$\frac{\mathrm{d}v}{\sqrt{1+v^2}} = \mathrm{d}x \Longrightarrow v(x) = \sinh x + c_1$$

con c_1 costante da determinare. Ora la prima equazione puo' essere risolta per integrazione diretta:

$$u'(x) = \sinh x + c_1 \Longrightarrow u(x) = \cosh x + c_1 x + c_2$$

con c_2 costante da determinare. Per trovare tali costanti basta far sì che vengano soddisfatte le condizioni ai bordi e risolvere dunque il sistema:

$$\begin{cases} u(-1) = \cosh(-1) - c_1 + c_2 = 1 \\ u(1) = \cosh(1) + c_1 + c_2 = 1 \end{cases} \Longrightarrow \begin{cases} c_1 = 0 \\ c_2 = 1 - \cosh(1) \end{cases}$$

in conclusione la soluzione del nostro BVP è:

$$u(x) = \cosh x + 1 - \cosh(1)$$

ora facciamo due piccole osservazioni che ci serviranno dopo: la prima è che $u'(-1) = v(-1) = \sinh(-1)$, che useremo poi come dato iniziale per la nostra integrazione numerica, la seconda è che tale soluzione è unica e la $u \in C^\infty(\mathbb{R})$.

Soluzione con BVP4C

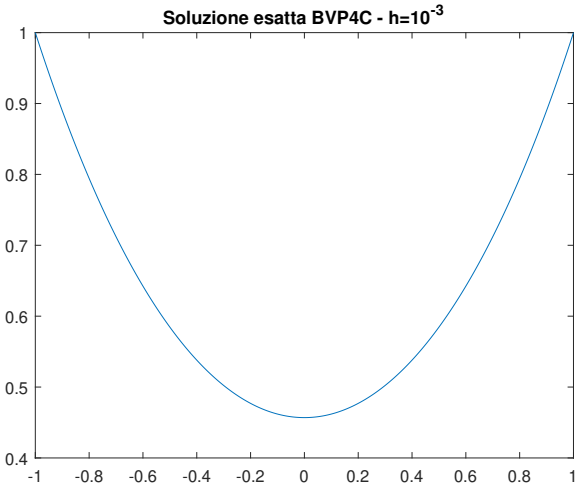
Tale funzione di MATLAB richiede tre (opzionalmente 4, contando le *options*) argomenti: l'equazione standardizzata, le condizioni al bordo e un *guess* iniziale della soluzione. Per le prime due si è già detto tutto nel paragrafo precedente. Per il *guess* abbiamo optato per la funzione costante $f(x) = 1$, tuttavia un'altra scelta possibile, irrisoriamente più efficace, sarebbe potuta essere la funzione $f(x) = \cos(x^2 - 1)$, ma abbiamo scartato tale opzione perchè di fatto non apportava nessun miglioramento tangibile ed aumentava solo il costo computazionale.

Options

Tramite il comando *bvpset* è possibile settare la tolleranza relativa (del residuo ad ogni passo di integrazione) e la tolleranza assoluta del comando *bvp4c*. Di default sono, rispettivamente, 10^{-3} e 10^{-4}. Noi abbiamo deciso di settarle rispettivamente a 10^{-4} e 10^{-10}. Purtroppo non è possibile scendere sotto a quel valore per la tolleranza relativa, siccome, altrimenti, il comando *bvp4c* riadatta la mesh (eventualmente aggiungendo nodi e/o spostandoli) per soddisfare la nostra richiesta. Per noi ciò è controproduttivo siccome rende poi molto difficile calcolare l'errore tra la soluzione ottenuta con *bvp4c* e quella ottenuta per integrazione numerica con il nostro script. Abbiamo inoltre attivato le *Stats*, in modo da vedere eventualmente quando la mesh viene alterata e quanto vale il residuo massimo. Fortunatamente il residuo massimo con i nostri settaggi risulta dell'ordine di 10^{-6}.

Grafico della "soluzione esatta"

A seguire il grafico ottenuto con *bvp4c* con una mesh con nodi equidistanziati e passo pari a 10^{-4}:



Discretizzazione alle differenze finite centrate

Dall'equazione sappiamo che, se esiste, la soluzione è sicuramente almeno di classe $C^2((-1,1))$ (in realtà sappiamo che è $C^\infty(\mathbb{R})$), dunque ha senso discretizzarla usando la tecnica delle differenze finite con passo costante pari ad h. In tal caso, come ben sappiamo, diventa:

$$\frac{u(x+h)-2u(x)+u(x-h)}{h^2} = \sqrt{1+\left(\frac{u(x+h)-u(x-h)}{2h}\right)^2}$$

Mettendo ora $u(-1)=u_0$ e $u(-1+n\cdot h)=u_n$ ci riconduciamo ora al seguente problema discreto sulla nostra mesh di $(N+1)$-punti equidistanti h:

$$\begin{cases} \dfrac{u_{n+1}-2u_n+u_{n-1}}{h^2} = \sqrt{1+\left(\dfrac{u_{n+1}-u_{n-1}}{2h}\right)^2} & \forall n = 1,2,\cdots,N-1 \\ u_0 = u_N = 1 \end{cases}$$

Tale problema è ovviamente non lineare, quindi per linearizzarlo approssimeremo la derivata prima di destra utilizzando il metodo di Newton. In tal modo otterremo un vettore colonna che chiameremo (in linea con la standardizzazione della prima pagina) $\underline{v} = (v_1,...,v_N)^T$. Ovviamente ci servirà un valore v_0 di innesco per il metodo di Newton, che sarà preferibilmente vicino al valore $u'(-1)=v(-1)$, siccome lo sappiamo, lo metteremo uguale a $v(-1)=\sinh(-1)$. Praticamente la funzione su cui applichiamo il metodo di Newton è:

$$v_n = v_{n-1} - \frac{v_{n-1}-v_0-h\sqrt{1+v_{n-1}^2}}{1-\dfrac{hv_{n-1}}{\sqrt{1+v_{n-1}^2}}}$$

ora costruiamo il vettore $\underline{f}$ dei termini noti del nostro sistema lineare, ricordandoci delle condizioni al bordo nei nodi 1 e $N+1$, e la matrice A di tipo $(N-1)\times(N-1)$ che deriva dall'equazione alle differenze finite centrate. Il tutto dunque risulta siffatto:

$$A\cdot\underline{u} = \frac{1}{h^2}\begin{pmatrix} -2 & 1 & 0 & \cdots & & \\ 1 & -2 & 1 & 0 & \cdots & \\ 0 & 1 & -2 & 1 & 0 & \cdots \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ \cdots & \cdots & \cdots & \cdots & 0 & 1 \quad -2 \end{pmatrix}\cdot\underline{u} = \begin{pmatrix} \sqrt{1+v_2^2}-\dfrac{1}{h^2} \\ \sqrt{1+v_3^2} \\ \vdots \\ \sqrt{1+v_N^2}-\dfrac{1}{h^2} \end{pmatrix} = \underline{f}$$

tale sistema sarà risolto da MATLAB tramite il comando $A\backslash\underline{f}$ e la soluzione andrà orlata con le boundary conditions. Questa implementazione si trova nel file *SistLin.m*.

Integrazione con metodo alternativo

Ci siamo accorti che in realtà il problema discreto, una volta linearizzato con Newton, può essere risolto in maniera differente. In particolar modo richiamando la standardizzazione del primo paragrafo:

$$\begin{cases} u'(x) = v(x) \\ v'(x) = \sqrt{1 + (v(x))^2} = f(v(x)) \end{cases}$$

ma ora, grazie a Newton, abbiamo il vettore $\underline{v}$ che, aggiungendoci $v_0 = \sinh(-1)$, contiene il valore di $v(x)$ in tutti i nodi della mesh; dunque, la prima equazione è facilmente integrabile con un qualsiasi metodo Runge-Kutta, ad esempio con Eulero Esplicito nel modo seguente:

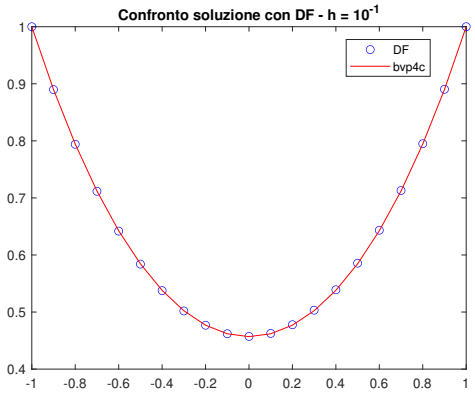
$$\begin{cases} u_n = u_{n-1} + hv_n \quad \forall n = 1, 2, \cdots, N-1 \\ u_0 = u_N = 1 \end{cases}$$

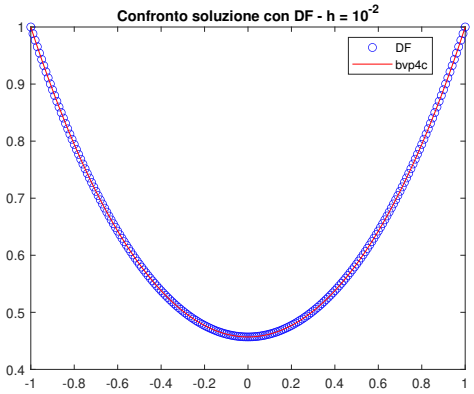
Adottando tale approccio si ha che l'algoritmo richiede un minor costo computazionale (ci sono meno valutazioni di f e non utilizza alcun algoritmo per risolvere sistemi lineari), a scapito di un ordine di grandezza dell'errore maggiore (ma questo è facilmente ovviabile implementando un metodo RK di ordine maggiore). Tale soluzione si trova nel file *EuleroEsplicito.m*.

Analisi dell'errore

Confronto tra soluzione esatta e approssimata

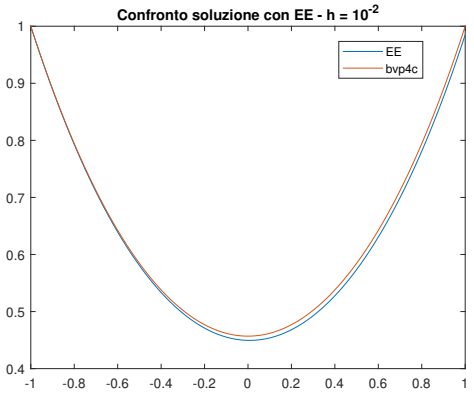
Metodo delle differenze finite





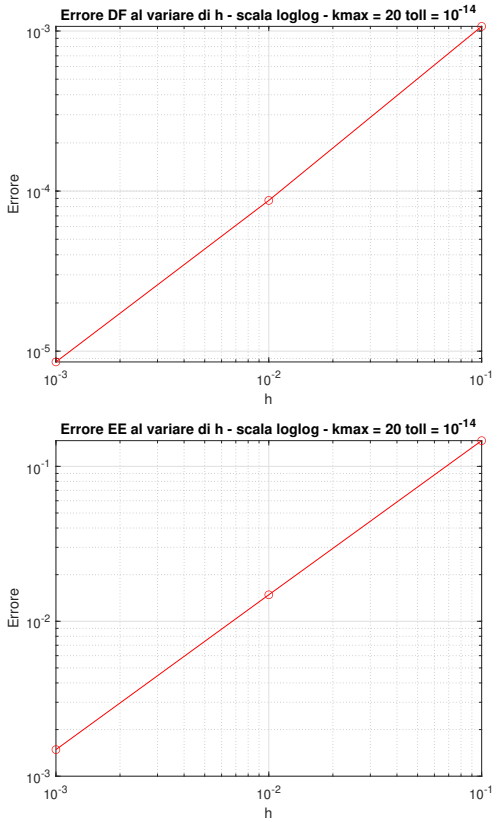
Si noti come già con $h = 10^{-2}$ la soluzione esatta e quella approssimata con le differenze finite centrate (con Newton con numero di iterazioni massimo pari a 20 e tolleranza pari a 10^{-14}) risultino indistinguibili senza ingrandimento.

Eulero Esplicito



Errore al variare del passo h

Nel primo paragrafo abbiamo osservato che $u \in C^{\infty}(\mathbb{R})$, dunque i metodi convergono con ordine pari all'ordine del metodo; infatti, facendo dei grafici in scala logaritmica su entrambi gli assi con $h = 10^{-1}, 10^{-2}$ e 10^{-3} si ottiene che l'errore (in norma infinito) si comporta come $O(h^2)$ per il metodo delle differenze finite e come $O(h)$ per il metodo di Eulero Esplicito. (Tenendo i parametri di Newton sempre pari a 20 come numero di iterazioni massimo e tolleranza pari a 10^{-14}).



Errore al variare dei parametri di Newton

Chiamiamo d'ora in poi $kmax$ il numero di iterazioni massimo del metodo di Newton e $toll$ la tolleranza. Nel codice MATLAB abbiamo fatto variare sia $kmax$ che $toll$, non abbiamo riscontrato differenze significative, in particolare abbiamo notato che basta già un $kmax = 5$ per far sì che il metodo converga anche con $toll = 10^{-14}$. L'unica osservazione significativa è che $toll$ deve essere minore o uguale ad h^2 (nel caso delle DF) o ad h (nel caso di EE) per far sì che i due metodi abbiano ordine di convergenza pari all'ordine del metodo.

Commento a posteriori sulla scelta di v_0

Se non avessimo saputo la soluzione esatta e il valore di $v(-1)$ avremmo potuto fare un guess iniziale e raffinarlo con il metodo di Newton. Riflettendoci un attimo un *educated guess* sarebbe dovuto essere un numero negativo siccome abbiamo che $u(-1) = u(1)$ e che $u'' > 0$ sul nostro intervallo, di conseguenza sicuramente la derivata prima in -1 sarebbe dovuta essere negativa.

Problema 2

Soluzione analitica

L'equazione caratteristica associata al problema è:

$$(\star) - D\frac{\partial^2 c}{\partial x^2} + u\frac{\partial c}{\partial x} + Kc = M \implies -D\lambda^2 + u\lambda + K = 0$$

le cui radici sono:

$$\lambda_+ = \frac{u + \sqrt{u^2 + 4KD}}{2D} > 0 \text{ e } \lambda_- = \frac{u - \sqrt{u^2 + 4KD}}{2D} < 0$$

dunque la soluzione dell'omogenea associata è del tipo $c_1 e^{\lambda_+ x} + c_2 e^{\lambda_- x}$ con c_1 e c_2 costanti da determinare. Visto il modello fisico in questione risulta sensato porre che i limiti del campo di concentrazione ad infinito siano zero; infatti, ci aspettiamo che nella zona a monte la concentrazione sia nulla (siccome la corrente va verso valle), mentre a valle, siccome $K > 0$, poniamo che definitivamente il fiume riesca a ripulirsi. Dunque le condizioni ai limiti di $c(x)$ risultano:

$$\lim_{x \to +\infty} c(x) = 0 = \lim_{x \to -\infty} c(x)$$

Notiamo che, necessariamente il problema va risolto su due intervalli disgiunti, rispettivamente $(-\infty, 0)$ e $(0, \infty)$. La soluzione dovrà essere comunque prolungata in 0 in modo C^0 per ottenere la soluzione su $\mathbb{R}$ con il valore $\frac{M}{\sqrt{u^2+4KD}}$. Risulta dunque essere:

$$c(x) = \begin{cases} \dfrac{Me^{\lambda_+ x}}{\sqrt{u^2 + 4KD}} & x \leq 0 \\ \dfrac{Me^{\lambda_- x}}{\sqrt{u^2 + 4KD}} & x \geq 0 \end{cases}$$

Discretizzazione alle differenze finite

Prima di discretizzare l'equazione $(\star)$ abbiamo diviso il tutto per D, ottenendo i parametri normalizzati $\tilde{u} = \frac{u}{D}$ e $\tilde{K} = \frac{K}{D}$. Dopodichè, come da richiesta, abbiamo posto $K = 1$. Per discretizzare la derivata prima e seconda abbiamo usato per tutti i nodi la tecnica delle differenze finite centrate per la derivata seconda e *backward* per la derivata prima, tolto che per l'ultimo nodo dove abbiamo usato le differenze finite *backward* anche per la derivata seconda. Si è deciso di modellizzare M come una gaussiana con media in zero del tipo:

$$M(x) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{x^2}{2\sigma^2}}$$

ed abbiamo chiamato M_n la valutazione di M nel nodo n-esimo della mesh. Ci siamo accorti che i risultati migliori si ottengono ponendo σ^2 due ordini di

grandezza minori dell'ampiezza del passo h; infatti, per σ^2 maggiori abbiamo che la sorgente non approssima bene una sorgente puntiforme, mentre per valori minori il metodo numerico si accorge di M solo per un numero molto ridotto dei punti della mesh. Dunque l'equazione diventa, una volta svolti i conti, su una mesh di $(N+1)$ punti con passo costante h:

$$(\star\star)\begin{cases} -\frac{1}{h^2}c_{n+1} + \left(\frac{2}{h^2} + \frac{\tilde{u}}{h} + \tilde{K}\right)c_n - \left(\frac{\tilde{u}}{h} + \frac{1}{h^2}\right)c_{n-1} = M_n \quad \forall n < N \\ \left(\frac{\tilde{u}}{h} + \tilde{K} - \frac{1}{h^2}\right)c_N + \left(\frac{2}{h^2} - \frac{\tilde{u}}{h}\right)c_{N-1} - \frac{1}{h^2}c_{N-2} = M_N \end{cases}$$

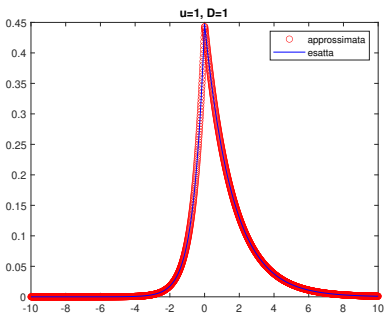
Come intervallo abbiamo scelto un intervallo aperto simmetrico del tipo $(-L, L)$. La scelta di L non è casuale, ma è stata fatta in modo da avere un intervallo sufficientemente ampio per far sì che resti sensata la condizione nel bordo sinistro $c(-L) = 0$ (cioè supponiamo di trovarci in una zona abbastanza a monte tale che l'acqua risulti pulita), per quanto riguarda il bordo destro abbiamo deciso di non mettere alcuna condizione siccome dipende dall'intensità u della corrente e dal coefficiente di diffusione D. Il nostro sistema si potrà dunque ancora una volta riscrivere nella forma:

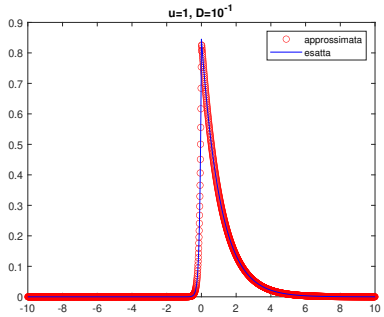
$$A \cdot \underline{c} = \underline{f} = \begin{pmatrix} M_1 \\ M_2 \\ \vdots \\ M_N \end{pmatrix}$$

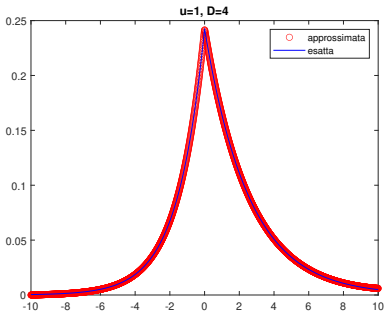
con A matrice $N \times N$ come suggerita da $(\star\star)$. Tale sistema sarà risolto da MATLAB tramite il comando $A\backslash\underline{f}$ e la soluzione andrà orlata con la boundary condition. Questa implementazione si trova nel file *SistLin2.m*.

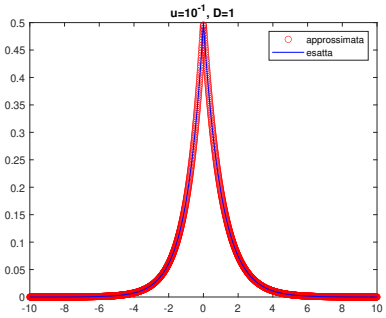
Confronto soluzione esatta e approssimata

Tutti i grafici seguenti sono ottenuti ponendo i seguenti valori ai parametri: $K = 1$, $h = 10^{-2}$, $L = 10$ e $\sigma^2 = 10^{-4}$.

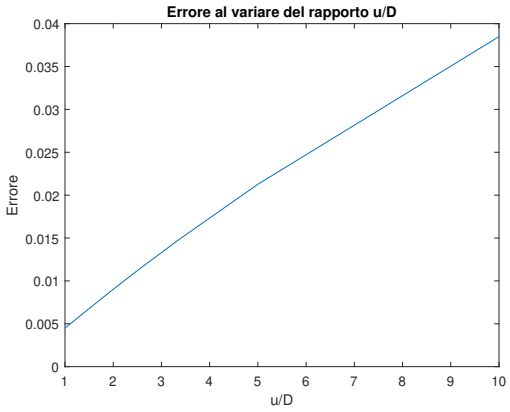








Notiamo che per valori di D molto piccoli la soluzione è molto *stiff* in un intorno di zero, per migliorarla si potrebbe usare una mesh con i nodi non equidistribuiti, proporremmo di usare una mesh con i nodi distribuiti in modo che l'integrale della gaussiana M su ogni intervallino sia costante. Con tale accorgimento avremmo i nodi concentrati in un intorno di zero e poi più distanziati man mano che ci spostiamo verso le code. Il grafico successivo illustra come, tenendo fisso $u = 1$, facendo variare D tra 10^{-1} e 1 mantenendo inalterati gli altri parametri vari l'errore.



Notiamo che più tale rapporto è prossimo ad 1, migliore è l'approssimazione.

Un nuovo problema

Soluzione esatta

Analizziamo il problema per $K = 0$. L'equazione caratteristica del problema è:

$$-D\frac{\partial^2 c}{\partial x^2} + u\frac{\partial c}{\partial x} = M \implies -D\lambda^2 + u\lambda = 0$$

dunque la soluzione è del tipo: $c_1 e^{\frac{u}{D}x} + c_2$, con c_1 e c_2 costanti da determinare. Ora cerchiamo la soluzione su un intervallo $(-L, L)$ con L sufficientemente grande da poter assumere la condizione nell'estremo sinistro sia $c(-L) = 0$, sensata

siccome la corrente va verso destra ($u > 0$). Notiamo ora che, siccome $K = 0$ (e quindi il fiume non si ripulisce da solo) non avremo più la condizione limite che il campo di concentrazione sia zero per $x \to +\infty$, invece avremo che dopo $x = 0$ dovrà restare costante, tale costante sarà pari alla quantità di inquinante sulla velocità della corrente, quindi nel nostro caso sarà $\frac{1}{u}$. Dunque la soluzione esatta sarà:

$$c(x) = \begin{cases} \dfrac{e^{\frac{ux}{D}}}{u\left(1 - e^{-\frac{Lu}{D}}\right)} - \dfrac{e^{-\frac{Lu}{D}}}{u\left(1 - e^{-\frac{Lu}{D}}\right)} & x \in (-L, 0) \\ \dfrac{1}{u} & x \in [0, L) \end{cases}$$

Discretizzazione alle differenze finite

La sorgente è stata modellizzata sempre come una gaussiana, esattamente come nei paragrafi precedenti. Abbiamo utilizzato le differenze finite centrate per calcolare la derivata seconda, mentre per la derivata prima abbiamo preferito usare le differenze finite *backward*, siccome la corrente va da sinistra verso destra, in tal modo restiamo *upwind*. Per l'ultimo nodo abbiamo usato le differenze finite *backward* per la derivata seconda e posto la derivata prima uguale a zero (perchè il campo è costante). Dunque, sempre usando una mesh di $(N + 1)$ punti con passo costante h:

$$(\star\star\star) \begin{cases} -\dfrac{D}{h^2}c_{n+1} + \left(\dfrac{u}{h} + \dfrac{2D}{h^2}\right)c_n + \left(-\dfrac{u}{h} - \dfrac{D}{h^2}\right)c_{n-1} = M_n \quad \forall n < N \\ -\dfrac{D}{h^2}c_N + \dfrac{2D}{h^2}c_{N-1} - \dfrac{D}{h^2}c_{N-2} = M_N \end{cases}$$

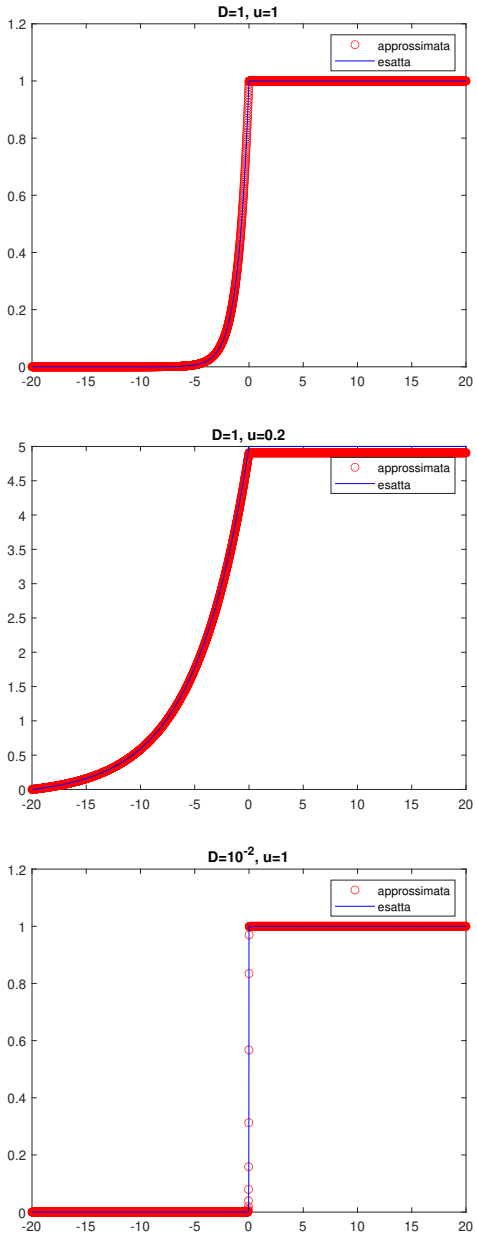
Il nostro sistema si potrà dunque ancora una volta riscrivere nella forma:

$$A \cdot \underline{c} = \underline{f} = \begin{pmatrix} M_1 \\ M_2 \\ \vdots \\ M_N \end{pmatrix}$$

con A matrice $N \times N$ come suggerita da $(\star\star\star)$. Tale sistema sarà risolto da MATLAB tramite il comando $A\backslash\underline{f}$ e la soluzione andrà orlata con la boundary condition. Questa implementazione si trova nel file *SistLin3.m*.

Confronto soluzione esatta e approssimata

Tutti i grafici seguenti sono ottenuti ponendo i seguenti valori ai parametri: $K = 0$, $M = 1$, $h = 10^{-2}$, $L = 20$ e $\sigma^2 = 10^{-4}$.



Notiamo che per valori di D piccoli il problema è molto *stiff* quindi potremmo provare, esattamente come sopra, ad infittire la mesh in un intorno di zero. Inoltre zoomando sulla parte destra del grafico della soluzione numerica ci si rende conto che in realtà non è propriamente costante, ma cio' dipende dall'errore di troncamento.

Interpretazione del modello fisico

Adimensionalizziamo il modello operando il seguente cambio di coordinate, indicando con c_0 il valore del campo di concentrazione in 0 e prendendo un intervallo di lunghezza $2L$:

$$\begin{cases} \alpha = \dfrac{c}{c_0} \\ \beta = \dfrac{x}{2L} \end{cases} \implies -\frac{\partial^2 \alpha}{\partial \beta^2} + \frac{2uL}{D}\frac{\partial \alpha}{\partial \beta} + \frac{4KL^2}{D}\alpha = \frac{2ML}{c_0}$$

Notiamo ora che $\dfrac{2uL}{D}$ ha la forma del numero di Peclèt e possiamo dunque vederlo come il rapporto tra l'inquinante trasportato dalla corrente e quello diffuso. Infatti da cio' possiamo ottenere le seguenti conclusioni:

- Se D cresce e u rimane fisso allora è più probabile che si inquini pure la zona a monte del fiume; inoltre, l'inquinamento del fiume sarà "simmetrico", cioè tanto a valle quanto a monte. Notare che valori negativi di D sarebbero alquanto insensati per il nostro modello.
- Se u cresce e D rimane fisso allora l'inquinante verrà trasportato più a valle e l'inquinamento del fiume si avrà per la maggior parte a destra del punto della sorgente di inquinante.
- Al crescere di M aumenta l'inquinamento totale del fiume e rappresenta l'inquinante in ingresso, è un getto impulsivo.
- Al crescere di K il fiume riesce a diminuire la quantità d'inquinante più facilmente, infatti con $K = 0$ si avrà che necessariamente l'inquinante dopo la sorgente verrà trasportato a valle senza cambiamenti in massa, dunque il campo di concentrazione sarà una retta costante con valore pari a $\dfrac{M}{u}$.